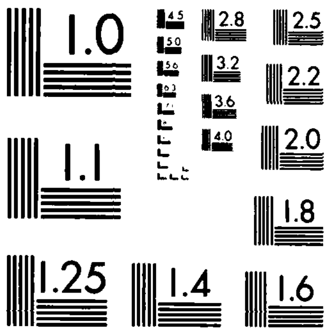
AD-A154 977 UW/NW (UNIVERSITY OF WASHINGTON/NORTHWEST) VLSI CONSORTIUM(U) WASHINGTON UNIV SEATTLE L SNYDER 18 MAR 85 MDA903-82-C-0424 1/1

UNCLASSIFIED F/G 9/5 NL

END

FILMED

DTIC


1.0
4.5
5.0
5.6
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6

(2)

AD-A154 977

UW/NW VLSI CONSORTIUM

Semiannual Technical Report No. 6

University of Washington

March 18, 1985

Reporting Period: 23 October 1984 to 18 March 1985

Principal Investigator: Lawrence Snyder

Sponsored by
Defense Advanced Research Projects Agency (DoD)
ARPA Order No. 4563
Issued by Defense Supply Service-Washington
Under Contract #MDA903-82-C-0424 (#MDA903-85-K-0072)

Contents

1 Executive Summary

1.1 Scope of This Report

This document reports on the research activities of the University of Washington/Northwest VSLI Consortium for the period 23 October 1984 to 18 March 1985 under sponsorship of the Defense Advanced Research Projects Agency. During that period, two contract numbers applied: MDA903-82-C-0424 and, after 21 January 1985, MDA903-85-K-0072.

1.2 Objectives

The goal of the UW/NW VLSI Consortium is to pursue research, development and education in computer aided design tools for VLSI systems with emphasis on technology transfer.

1.3 Accomplishments

During the reporting period, the Consortium concluded its work on the earlier contract and embarked on its new VLSI Design Generators Project. Highlights of both endeavors are:

1. Project Initiation. The Quality VLSI Design Generator Project began by setting the project's goals, defining near term plans, and building a technical foundation to serve as the basis of generator construction. This effort set the agenda for our project (See Section 2). We also participated in an internal Design Generator Workshop at Tektronix in order to exchange technical information between UW and the (three) projects underway at Tek.

2. Initial Design Generators. A VLSI Design generator packages expertise. In order to have expertise to package, we have chosen several circuit familys to implement. These designs, including RAM, ROM, Multipliers etc., have undergone several design iterations and have been packaged into *ad hoc*, i.e. not yet based on unified principles, generators.

3. CFL. In order to provide the software out of which generators can be built, we have developed a general, procedural-based layout description called Coordinate-free LAP. This facility, which can serve as an "offline" layout language, is the layout facility used in the current *ad hoc* generators.

4. Testing. The testing component of our work has featured two activities. A copy of the MIT Lincoln Labs non-contact laser probe was constructed. Also, we have been recalibrating our RNL simulation software to more accurately reflect the MOSIS fabrication characteristics. (See Section 5).

5. Education and Tape Distribution. The Consortium supported four UW VLSI Design classes during the reporting period: A testing class, an introductory class, a simulation class, and an advanced lab, as well as conducting a one day seminar for representatives of member firms. Our CMOS design system continues to be distributed and a new release is planned.

2 Introduction to VLSI Design Generators

The UW/NW VLSI Consortium is embarking on a multiyear project focused on VLSI design generators. The purpose of the project is to study the circuit generation with a view towards ensuring quality results, integrating generators into a full custom environment, and understanding the generator construction methodology. Our plan is to create a number of generators which we will use to create substantial VLSI designs.

2.1 A Description

A VLSI design generator is a program that, given a set of parametric specifications, produces (among other things) the layout for one member of a family of circuits. PLA generators are a well known type of design generator although they typically fail the standards of generality and flexibility which we will be trying to achieve. Specifically, a generator should be able to be flexible in terms of achieving various speed/area/power trade-offs, it should support various design regimes (e.g. domino or NORA), different pitches and wire placements, it should balance drivers to handle the loads of specific situations. Moreover, there are features that will be specific to each circuit family, e.g. adders can differ in the number representation employed, and flexibility must be provided here. In general, a generator packages expertise and the more ways a circuit can be customized to a situation, the more types of expertise can be applied.

2.2 Objectives of the Project

In the U.S. today, there must be dozens of "generator" *development* projects underway; there are few generator *research* projects. The purpose of the generator development efforts is to produce programs that can create circuits for a specific purpose, e.g. product family or solution methodology. Our research goals are many, but generally speaking, we are studying the generation process: Are generators like compilers or database management systems in that they tend to have common structure and components, independent of the specific instance? What are these components and how do they differ from instance to instance? How does a generator interface with a custom environment? Is it possible to integrate a collection of generators into a custom CAD system so that the complexity of checking and simulating the designs is proportional to the custom portion and not the total (custom and generated) size of the designs?

2.3 Structure of a Generator

In order to be more specific about the concept of a generator. We present a description of how a generic generator might be structured.

The generator process can be viewed as four coupled processes in which input parameters are provided and several results are produced:

- parameter validation
- model construction
- specialization
- generator

Among the outputs are a layout, interface to the DRC, interface to a simulator, etc. In parameter validation, the inputs are checked to be consistent and within range, and perhaps, optimized if appropriate. The model construction involves creation of the data structure(s) that characterize the different aspects of the circuit. The specialization routine particularizes the circuit for its specific application, e.g. sizing drivers, scaling wires, modifying pitch. Finally, the routines that produce the various outputs are evoked and the layout, net list, documentation, etc., are generated.

This view, which is idealized in the same way that compilers are idealized into lexical, syntactic and code generation phases, serves as a basis for our work. As our understanding increases, we will be able to better define these components.

2.4 Generators and Their Alternatives

Cell libraries and silicon compilers have been widely discussed as solutions to the same problem that our generators address: effective support of the VLSI designer. Our generators are more flexible than cell libraries which are based chiefly on macro substitution principles and thus are generally very rigid. Our generators are less general than a full silicon compiler because we wish not to compromise on the performance of our circuits, which is a possible consequence of generality. Our solution will require us to create a reasonable library of generators and it will require the designer to learn the details of the input parameters of each generator in order to use it analogously to using the TTL catalog.

However, by supporting common architectural components with high quality designs, we hope to give the designer the ability to produce high performance systems with a minimum of effort.

3 Initial Design Generators

The initial versions of ROM, RAM and multiplier generators have been completed. They are described in the appendices. Instances created by all generators are currently being fabricated by MOSIS. A few parts have been returned and are functional. When the remaining parts are returned, performance issues will be addressed. Modifications may be required of the generators or limits may be placed on their parameter ranges. The multiplier generator is currently being employed in a digital filter design.

A number of other generators are in various stages of development. These include a MUX, PLA and CAM. Nearly complete is a padframe generator that will accept a MOSIS standard frame as input and place pads in a user-specified orientation.

All the current generators are written with the aid of a package of C routines developed at the Consortium. This package, CFL (Coordinate Free LAP), allows cells created with the graphical editor Caesar to be assembled procedurally. All the facilities of the C language are available for the manipulation of the generator's input data and construction of the layout. CFL is described in the following section.

4 Coordinate Free LAP

In the early stages of the construction of the first prototype generators, it became apparent that our existing procedural layout tools (PLAP in particular) were inadequate for the construction of large parametrized layouts. The logistics of manipulating large numbers of cells with Pascal procedures proved unmanageable. Out of this experience came the development of a new tool, Coordinate Free LAP (CFL).

CFL is a library of routines written in C. It allows the designer/programmer to specify placement of cells relative to each other, (hence "coordinate free"). Additionally there is a coordinate dependent facility called "wire" for generating arbitrary configurations of material, whenever that is more convenient.

The system has been designed to operate in conjunction with the Berkeley layout editor Caesar in that it is capable of both accessing symbols generated using Caesar and writing symbols which may be accessed by Caesar. Although CFL has sufficient functionality to allow definitions to be developed for all artwork including the lower level cells in a design, it is intended to be used more in the mode of chip assembly. Hence the typical application would involve using Caesar to generate lower level cells or tiles and then CFL facilities for assembling these leaf cells into higher level modules.

To insure that a wide variety of assembly situations can be accommodated, CFL includes approximately 70 variants of operators for juxtaposing, transforming, and replicating hierarchies of symbols. The syntax of these operators is quite compact since generated symbols are simply stored in program variables of type SYMBOL*.

All of the calculations which support the operators of CFL are performed from border descriptions of the borders of the symbols. The information in the border descriptions includes the bounding box and lists of coordinates of the points where each kind of material in the symbol makes contact with the bounding box. In this way, modules which have a large number of rectangles may be accessed from the library without the need of reading all of the files associated with their sub-modules. This capability allows CFL to assemble large blocks of circuitry extremely quickly. CFL also provides automatic hierarchy compression when symbols are written to disk so that only those symbols which represent meaningful functional groups need be saved.

5 Experimental Facilities

The project is processing a significant number of CMOS chips. As a result, testing activities have taken on a greater significance.

5.1 Laser Probe Experimentation

Non-contact probing of integrated circuits is a valuable tool for troubleshooting prototype designs. Such a tool, combined with a circuit simulator, is an ideal instrument for locating design flaws. Since the complexity of circuit designs is increasing rapidly, it is more and more important to sense a voltage or a logic state on a circuit node having dimensions given by minimum design rules. Traditional wafer probers cannot reliably access such nodes within the circuit periphery.

The laser prober prototype presented by MIT Lincoln Labs at an earlier DARPA Contractors meeting appeared to provide a simple, reliable means to access internal nodes without potential damage to the unit under test. With their help in supplying key custom parts and instructions for procurement of the others and testing, we have been able to duplicate that capability. What remains to be done is to review all adjustments critically and to develop a laserspot location adjustment procedure so that no ambiguity remains in the determination of the state of a node being probed. At the current time we need more experience with the tool to reliably determine states by laser probing. We anticipate that our experience obtained while building this tool will suggest improvements for documentation so that this valuable capability will be easier to duplicate.

We acknowledge the support of Jack Raffel and Terry Hearndon of Lincoln Labs.

5.2 Simulation and Testing

Considerable effort has been expended to calibrate the RNL switch-level simulator with parameters returned by MOSIS. A derivation of RNL resistances from process parameters is presented in the appendix. Comparisons of simulations and measurements on actual chips have also been performed for a ring oscilator. As soon as fabrications are returned for large instances created by the multiplier and ROM generators, comparisons can be made to validate/invalidate performance predictions made by the simulators. Acquisition of additional testing equipment to complement the existing DAS 9100 will further refine the comparisons.

6 Educational Activities

On February 21, 1985, the Consortium presented an all-day program of lectures to a group of representatives from Consortium member firms. This yearly program recounts the accomplishments of the past year and describes future directions. Included in this year's program were a number of guest lecturers speaking on VLSI education and trends in computer aided design. A large portion of the program was devoted to technical talks by Consortium staff members. The program was followed by an informal tour of the Consortium's VLSI laboratory. Over 60 attendees were present at this year's program.

The Consortium provides the facilities not only for the research described herein, but also for the University of Washington VLSI classes. In the reporting period, four classes have been supported:

- Introductions to VLSI Design (CS)
- VLSI Circuit Design and Simulation (EE)
- VLSI Chip Testing (EE)
- Advanced VLSI Design Lab (CS)

Each course makes extensive use of the facilities. In the Spring, there will also be a layout class supported by our equipment.

7 Distribution of CMOS Toolset

Release 2 of the Consortium CMOS toolset has been available since August 1984. Since that time 115 copies have been distributed to Universities and DARPA contractors. The release is supported under Berkeley UNIX 4.2. A complete listing of its contents has appeared in a previous technical report. A considerable amount of valuable feedback has been received on its use.

Another release of the toolset is planned for early summer. Included in this release will be enhancements for second layer metal and the package of routines referred to earlier as Coordinate Free LAP. Initial versions of several generators including the padframe generator will also be included.

A Layout Generator for a Static CMOS Multiplier

Wayne Winder, UW/NW VLSI Consortium

Description:

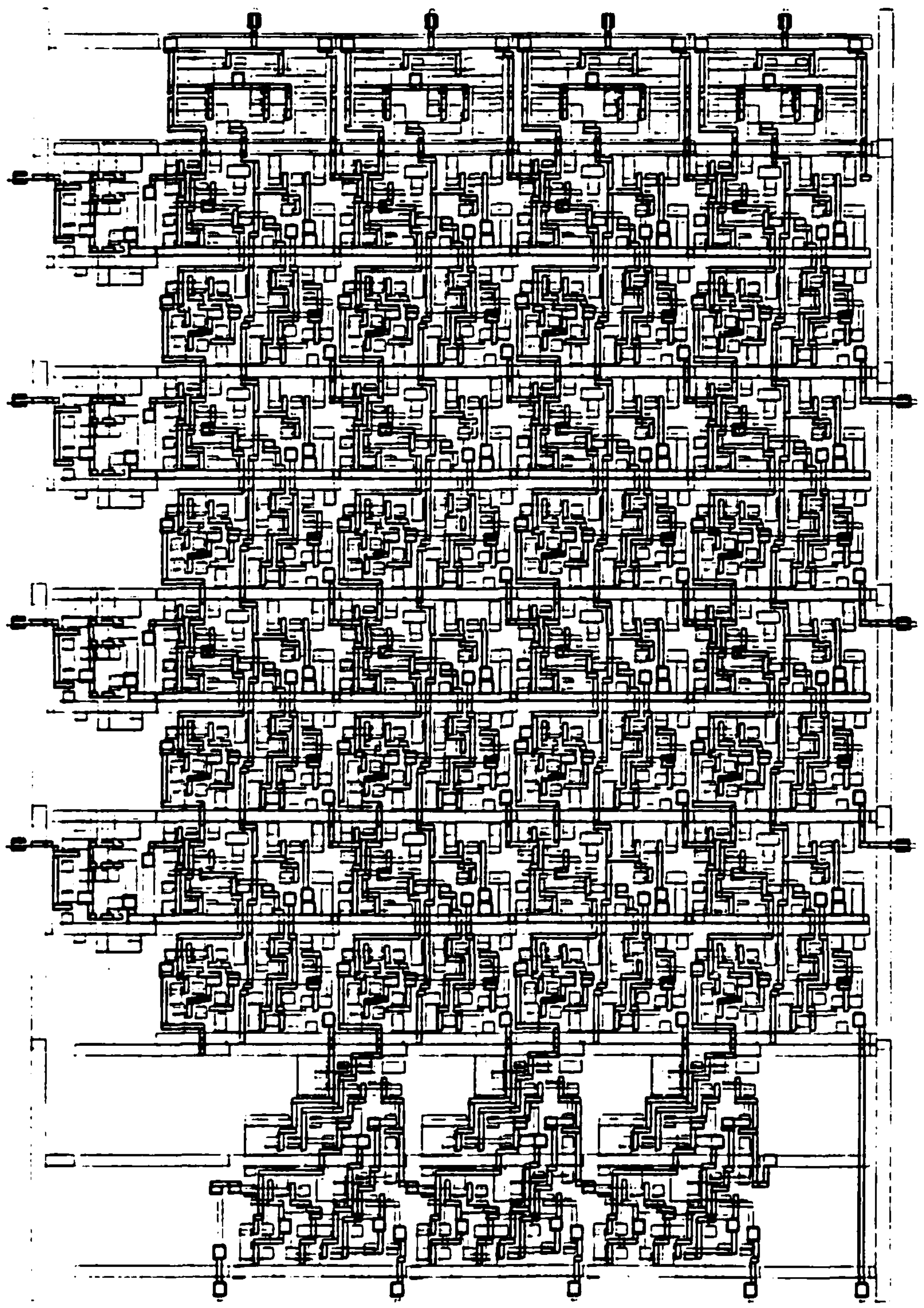
The multiplier generator we have developed was written to operate under the UW VLSI Tools environment. It is a "C" program using the CFL layout procedures described previously. The generator creates a geometric description in the "Caesar" data base format. This format is completely compatible with the rest of the UW VLSI Tools. The design implemented has been successfully fabricated through the MOSIS facility in 3 micron p-well CMOS.

The generated Multiplier modules operate on "m" bit multiplicand and "n" bit multiplier operands to create an "m+n" bit product. The multiplier module produced is rectangular. The multiplicand operand is on the top with the least significant bit to the right. The multiplier operand is on the left side with the least significant bit at the top. The product appears on the right and bottom with the least significant n-1 bits to the right (least significant bit on top) and the most significant m+1 bits on the bottom (most significant bit to the left).

The generator creates a basic multiplier array composed of static partial product/adder cells in an m by n array. The sumout signals of these cells propagate one row down and one column to the right while the carryout signals propagate one row down. The partial product bits are formed from the multiplicand and multiplier operand signals which are buffered to provide the signals over the width and depth of the multiplier, respectively. The high order m+1 bits of the product are formed by an m+1 bit adder across the bottom of the multiplier module. Any style of adder is possible, a ripple adder was implemented for simplicity and space conservation. Driver buffers are added on the left and across the top. Connections are then provided to vertical ground and power buses on the left and right sides of the multiplier module. Wiring connections to the product, multiplicand and multiplier operands are added and labeled.

The design approach for the multiplier generator is the leaf-cell assembly approach. Leaf cells are generated with a graphical editor ("Caesar"). These cells are then manipulated procedurally, using "CFL", to create the multiplier circuitry. There are a total of 31 leaf cells which vary in size from cells with 3 rectangles to cells with 18 transistors. Many of the cells are quite similar.

A Generated 4 x 4, Unsigned Multiplier



Options:

The initial set of options is limited in order to concentrate on the functionality of the circuit:

1. m and n - bit widths of the multiplicand and multiplier operands, respectively.
2. number representation - unsigned and two's complement integer representations are supported.
3. Vdd or Ground on the left - the vertical buses on the left and right of the multiplier may be Vdd/Ground or Ground/Vdd.
4. labels for multiplicand, multiplier and product - the user may specify the labels for the inputs and outputs of the multiplier circuitry. These labels are appended with 0, 1, ... for each bit, from the low order bit up.

In addition, the multiplier generator automatically sizes the multiplicand and multiplier buffers for the signals which carry across the width and depth of the array. The generator also computes the required widths of the vertical power and ground buses. The sizes and widths are computed from specific process parameters.

Results:

An 8 by 8 unsigned multiplier is generated in about 10 seconds on the UW VLSI Vax 11/780. A signed multiplier of the same size takes about 20 seconds. A 16 by 16 signed multiplier takes about 35 seconds.

The approximate size of an 8 by 8 multiplier (signed or unsigned) in the 3 micron MOSIS technology is 1050 microns by 1400 microns. There is close to linear growth in each dimension for greater sizes.

The 8 by 8 multipliers operate in about 150 nanoseconds. More accurate times are not yet determined. The delay grows approximately linearly with the sum m+n. More accurate measures are available on the ripple adders implemented. The successfully fabricated 7 bit ripple adders measured gave delays in the range 88 to 99 nanoseconds.

A Dynamic ROM Generator

David J. Morgan

Design Style

Dynamic

Dynamic designs give parts which are faster, use less power, occupy less die area, and don't require ratioed transistors for correct operation. The major reason for this is that each boolean operation requires a single transistor for each input, versus two for static designs. Reducing the number of transistors reduces interconnect and gate capacitance, giving smaller, faster designs. Figure 1 illustrates this concept for a 3 input NOR gate.

Another reason for faster operation with less power is that nodes are never driven in two directions at the same time. All current serves to charge capacitance rather than generating heat.

Dynamic designs are not without problems (clock skew, noise rejection and charge sharing) but the savings make them well worth the effort.

Since ROMs are typically large structures we need a design which minimizes the tendency to slow down for large instances. Using the NOR form (transistors in parallel) for decoders and bit plane satisfies this need.

This generator implements two dynamic design styles with NOR forms. Simplified schematics of each each design style are shown in Figures 2 and 3. The Multiple Clocks version has N-channel decoders and bit plane but requires multiple clock phases. The NORA[1] version has P-channel decoders, N-channel bit plane, and one clock phase. The reason for a single clock phase when two are shown, is that they can overlap; this allows one of them to be generated with an inverter from the other. The Multiple Clocks version is susceptible to clock skew but only requires a single metal layer. The NORA version is slightly larger and requires two metal layers.

APPENDIX A: MOSFET Model

Farad = Coulomb/Volt Amp = Coulomb/Second

		UNITS
ϵ_o	= permeability of free space = 8.85 exp-12	*Farads/Meter*
ϵ_{si}/ϵ_o	= Relative permeability silicon = 11.7	*none*
$\epsilon_{sio2}/\epsilon_o$	= Rel. perm. of silicon dioxide = 3.9	*none*
T_{ox}	= Transistor gate oxide thickness	*Meters*
C_{ox}	= Transistor gate capacitance/area=ϵ_{sio2}/T_{ox}	*Farads/Meters*^{2}
q	= Charge of electron = 1.602 exp-19	*Coulombs*
N	= Dopant Density	*atoms/Meter*^{3}
γ	= body or back gate factor $=((2 * \epsilon_{si} * q * N)^{0.5})/C_{ox}$	*Volts*$^{0.5}$
μ	= Carrier mobility	*Meters*^{2}/Volts* Seconds
K_p	= Transconductance parameter $=\mu * C_{ox}$	*Amps/Volt*^{2}
n_i	= Density of carriers in intrinsic material	*atoms/Meters*^{3}
ϕ_f	= Surface Potential $=(k * Temperature/q) ln(N/n_i)$	*Volts*
V_{sb}	= Transistor source to substrate voltage	*Volts*
V_{to}	= Transistor zero bias threshold voltage	*Volts*
S	= Transistor shape factor = transistor width/heigth	*none*
V_{th}	= Transistor threshold voltage $= V_{to} + \gamma * ((V_{sb} + 2 * \phi_f)^{0.5} - (2 * \phi_f)^{0.5})$	*Volts*
δ	= Channel charge distribution correction factor $=(\gamma/2) * (V_{sb} + 2 * \phi_f)^{(-0.5)}$	*none*

To determine a transistor's static resistance it is necessary to calculate its instantaneous resistance with the terminal conditions present when it is "fighting" another transistor. In an NMOS design this would be with the enhancement transistor in its linear regime. The depletion device's regime of evaluation would be dependent on its threshold voltage (V_{th}) so this is determined first.

For CMOS designs the calculations are dependent on the design style you are using. For ratioless designs I would suggest calculating the p-channel and n-channel static resistances in their saturated regimes at $V_{ds} = \frac{V_{supply}}{2}$. This will cause generation of a ratio error if your design depends on the relative values of the transistors to give a valid logic level, which should not be the case for a ratioless design. On the other hand if you have a ratioed design where the p-channel transistor is being used as a substitute for the depletion transistor in an NMOS design, then you would probably want to evaluate the p-channel in its saturated regime, and the n-channel in its linear regime, with the appropriate conditions when the pulldown transistor is turned on.

A program has been created to calculate a transistor's RNL resistances. It allows you to specify the change in a node's charge in terms of absolute voltage, as a fraction of supply voltage, and as a fraction of the V_{th} . The data structures have been designed to allow resistance values at different process corners and load configurations to be easily accomodated. This program currently requires you to enter process constants directly into the source code, recompile, and run to obtain new resistance values.

Appendix B contains the integrals calculated by this program.

The resistance values mentioned in Figure 1 are not really the ohms which we are familiar with. To simplify calculations in RNL a scaling factor is included in this value such that multiplying it by the capacitance of the node it is driving gives the time required to change the charge on that node. In RNL the change in charge is specified with V_{hi} and V_{lo} . To simplify this concept let us demonstrate it for a simple resistor.

$I = Current \quad C = Capacitance \quad V = Volts \quad T = Time$

$$I = C(\frac{dV}{dT})$$

or

$$dT = C(\frac{dV}{I})$$

Integrating both sides: $T = C \int \frac{dV}{I}$

For a resistor: $I = \frac{V}{R}$

This gives:
$$T = RC \int \frac{dV}{V}$$
$$= RC\,(ln(V_{hi} * V_{sup}) - ln(V_{lo} * V_{sup}))$$
$$= RCln(\frac{V_{hi}}{V_{lo}})$$

For $V_{hi} = 0.9$ and $V_{lo} = 0.1$, Transistion Time $\cong 2.2RC$.

The Resistance value used by RNL would be the resistor's actual value times 2.2 in this case (where change in charge is from $0.1V_{sup}$ to $0.9V_{sup}$). Transistor values are derived in the same way. The only complication is that the expression relating a transistor's voltage and current makes the integral cumbersome. A reasonable approximation for a transistor's voltage/current relationship are the modified Schichman-Hodges equations from L.Glasser's and D.Dobberpuhl's forthcoming book "The Design and Analysis of VLSI Circuits". Appendix A contains these equations and Appendix B shows their integration.

In addition to the equations relating current to voltage, limits of integration are required. As a practical matter a lower limit of $0.3 * V_{sup}$ and an upper limit of $0.8 * V_{sup}$ seem to give fairly accurate results for static CMOS designs. The values for accurate simulations of dynamic design styles may be quite different. The selection of appropriate limits is a subject worthy of separate treatment.

RNL Calibration

This document is intended to clarify the calculation of transistor resistance values used by RNL. This information should aid people who are interested in improving simulation accuracy for their designs.

RNL has three models for each transistor in a design. Two of these are used to calculate transition times (dynamic-high for rising, dynamic-low for falling), and a third (static) to determine a nodes steady state voltage when driven by two transistors simultaneously (as in ratioed NMOS logic).

The dynamic-high and dynamic-low models in RNL treat each transistor as a perfect switch in series with a resistor where the state of the switch is determined by the type of transistor, its gate voltage, and its RNL thresholds. Multiplying V_{hi} or V_{lo} by the supply voltage determines the gate voltages at which the switch state changes. For N-channel transistors any gate voltage above $V_{hi} * V_{sup}$ turns the transistor on and a voltage below $V_{lo} * V_{sup}$ turns the transistor off, values between these place the switch in an undefined state. Unique RNL thresholds (V_{hi} and V_{lo}) can only be assigned to each node, which means that all transistors whose gates share the same node have the same RNL thresholds. Figure 1 illustrates these principles for some typical P-channel and N-channel transistors.

Transistor	Vhi	Vlo	Vsup	Vgate	Load	Resistance
P-channel	.8	.3	5.0	1.0	Drain	20 KOhms
"	"	"	"	1.0	Source	200 KOhms
"	"	"	"	4.5	x	Very Large
N-channel	"	"	"	4.5	Drain	10 KOhms
"	"	"	"	4.5	Source	100 KOhms
"	"	"	"	0.5	x	Very Large

Figure 1.

The generator is being layed out using MOSIS 3 micron CMOS design rules. Caesar is the layout tool through which the leaf cells (decoder drivers, decoder slices, memory cells, i/o interfaces) are input. These will then be laminated using CFL, a procedural layout language.

The status of the project at present is as follows:

1. transistor level circuit design including SPICE simulation has been performed
2. topological (floor planning) is complete
3. leaf cell layout is very near completion

Left to do on the project is:

1. complete leaf cell layout
2. build a prototype by hand and submit it on a fab run
3. bind leaf cells into a generator using CFL
4. expand design to include algorithmic driver sising.

DUAL PORTED RAM GENERATOR

Barry Jinks, Microtel Pacific Research Liaison

Dual ported random access register files are useful for a wide variety of data path applications. Their utility stems from the need to present both inputs to an ALU simultaneously. Hence, a register file which allows coincident reading of data from two distinct addresses in the same address space eliminates the need for a latch and saves clock cycles.

Generic dual ported memories come in several varieties including those which allow simultaneous reading and writing of two addresses. In the version at hand, I have chosen to limit the operation to simulataneous reading from two addresses but write to only one – one of the addresses is ignored on a write operation. This tradeoff has been made because it allows the circuit to be simplified somewhat when compared with other varieties. As well, the majority of applications need only the simultaneous read feature. In short, the design philosophy used is an attempt to provide maximum functionality with minimum silicon.

The input parameters to the generator will be the following:

- the data word width
- the number of words required
- the number of inputs to the word line decoders
- the number of inputs to the bit line decoders

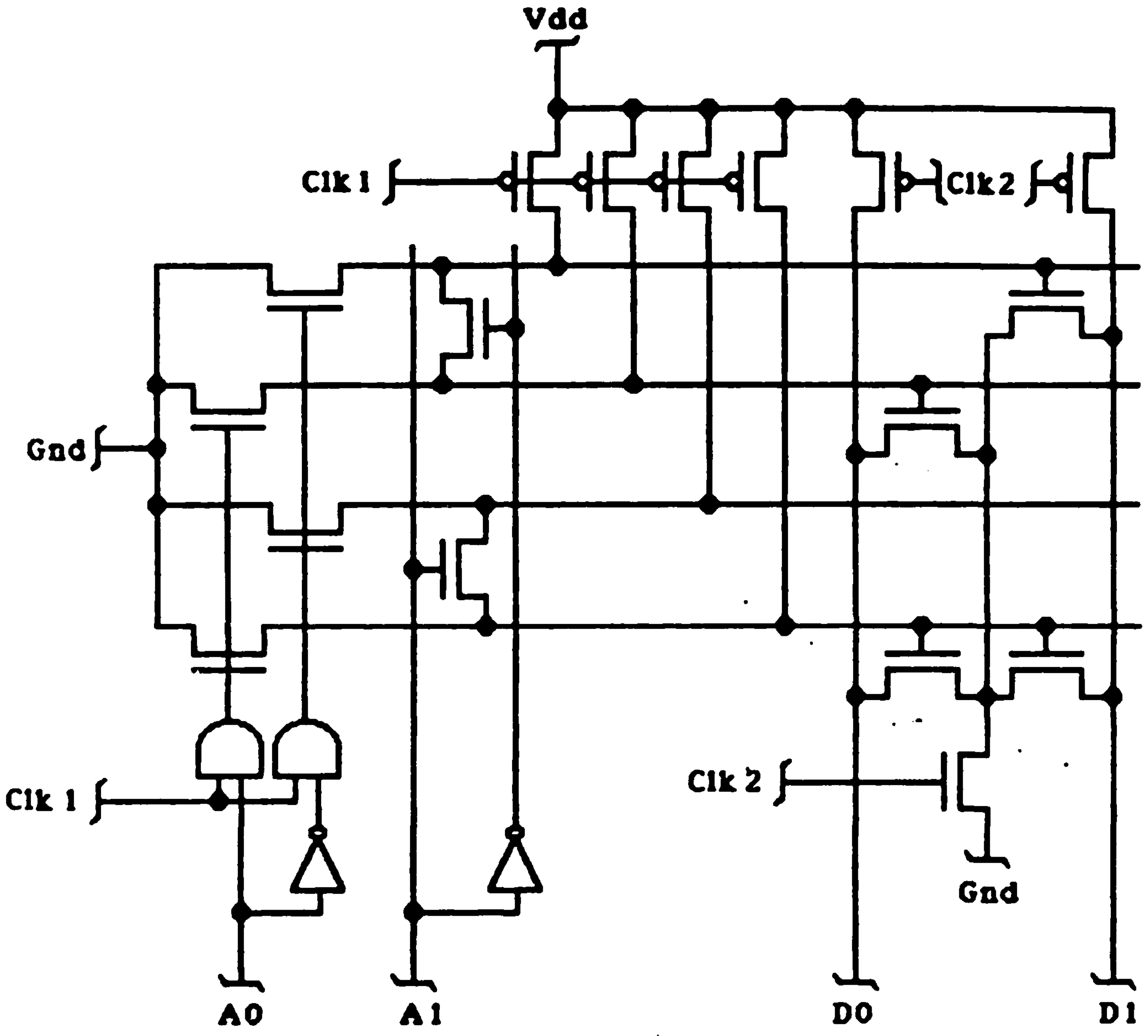
The topological requirements for generated circuit elements vary from project to project. For example, if the designer needs fast and compact memory he/she will probably choose a "stand alone" memory with both bit and word line decoders. On the other hand, a data path chip will need memory which can be configured as a slice which runs in the direction of the path. To address this, the designer will be given the choice of topology and path spacing in addition to the aforementioned input parameters.

The circuit design methodology being investigated for this generator is domino CMOS. The reasons for this are: (a) compared with static CMOS, dynamic designs are generally faster and (b) compared with other dynamic techniques (such as NORA), domino is more compact. The price paid is versatility since there is a restriction placed on timing. In the design at hand, address lines and the r/w line must become valid during precharge and must be held during the evaluate phase of the clock. In most designs this does not present a problem.

References

1. N.F.Goncalves, H.J.DeMan, "NORA: A Racefree Dynamic CMOS technique for Pipelined Logic Structures", IEEE Journal of Solid State Circuits, Volume SC-18, pages 261-266, June 1983.

Multiple Clocks 4x2 Bit ROM: (2 Phase)



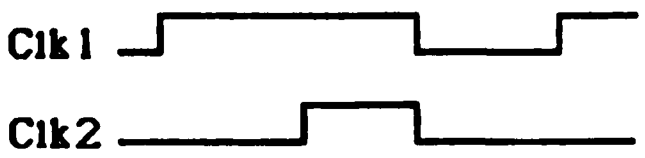


Figure 4

Improvements

Generators allow the designer to create quality designs faster than he might with a full-custom approach. The following improvements will increase this generators utility in fulfilling this goal:

Trade Area for Speed

There are two ways in which additional area can provide faster operation of generated ROMs. Dynamic sizing of drivers and self-timing circuitry. The latter makes the ROM appear static which may impact its integration with other funtional blocks on the chip.

Sense Amplifiers

These designs require bit lines to be discharged to less than a static inverters logic "0" threshold in order to detect the presence of a bit transistor. Adding sense amplifiers would allow us to trade noise margin for speed.

Poly Word Lines strapped with Metal-2

The poly word lines of these designs account for a large part of the total delay due to their distributed resistance and capacitance. By strapping this polysilicon with second layer metal this delay should be greatly reduced.

Two Phase Multiple Clocks Design

The dead times required for non-overlapping clocks can be eliminated by using two overlapping clocks as shown in Figure 4. This will allow faster operation with no power or real estate penalties.

Specifications

The generator specifications are as follows:

```
*Density
     2048x8 bit ROM in 3mm x 4mm, 3 micron MOSIS CMOS
     (size includes pad frame)

*Speed
     2048x8 bit ROM in 5 minutes CPU time.

*Design Style
     Multiple Clocks
     NORA

*Size and Shape
     Number of Bits = (BitsHigh) x (BitsWide) = 2**(n+k+m);
       BitsHigh = 2**n                          where n=1,2,3...
       BitsWide = (BitsOut) x (2**m)            where m=1,2,3...
       BitsOut = 2**k                           where k=1,2,3...
```

The designer specifies the shape as either wh6 (width/heigth=6) or hw4 (heigth/width=4), m and n are automatically selected to give the best fit.

*Programming Data
ASCII hex file or "random" for floor planning.

ROM Generator

Construction

This generator uses a set of "C" routines, written by Bill Beckett of the UW VLSI Consortium, and known as CFL (Coordinate Free Layout) . They allow caesar cells to be joined in a common caesar file with different allignment operators. As CFL manipulates a bounding box (created by CFL) rather than the entire cell it is extremely fast. The finished generator is a "C" program and a collection of caesar cells.

Creation of the generator involved the interpretation of an existing design (schematic and layout) into the caesar cells and routines mentioned above. In this case a 32x2 bit Multiple Clocks ROM was designed and interpreted. The NORA interpretation was obtained by modifying the Multiple Clocks generator.

Three man-months were required to produce the Multiple Clocks version, and one man-month to create the NORA style.

Validation

Validation of the generator involved DRC (design rule checking) and simulation of a number of instances, followed by fabrication and testing of the parts. DRC and simulation instances were selected to bracket the places where algorithms change for different structures (i.e. 2 bits out versus 4 bits out).

Fabrication and test results are as follows:

Style	Size	Status	Access Time	Yield
Mult. Clocks	32x2	Done	* < tacc < 1mS	5/5
Mult. Clocks	512x8	Testing	* < tacc < 1mS	?
Mult. Clocks	2048x8	Testing	360nS < tacc < 1mS	?
NORA	2048x8	Fabricate		

* Lower limit unresolved due to test equipment limitations.
? Awaiting further tests.

NORA 4x2 Bit ROM:

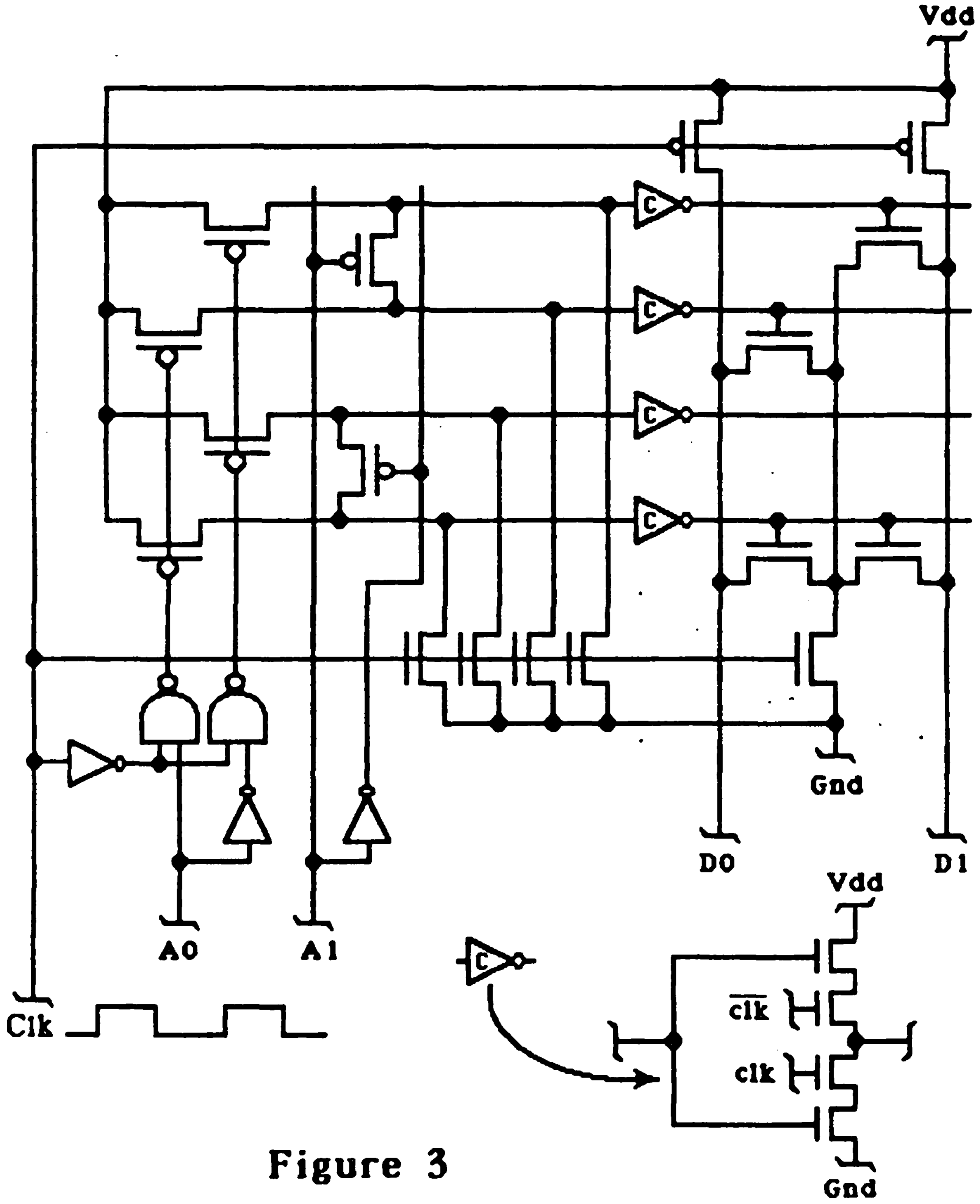
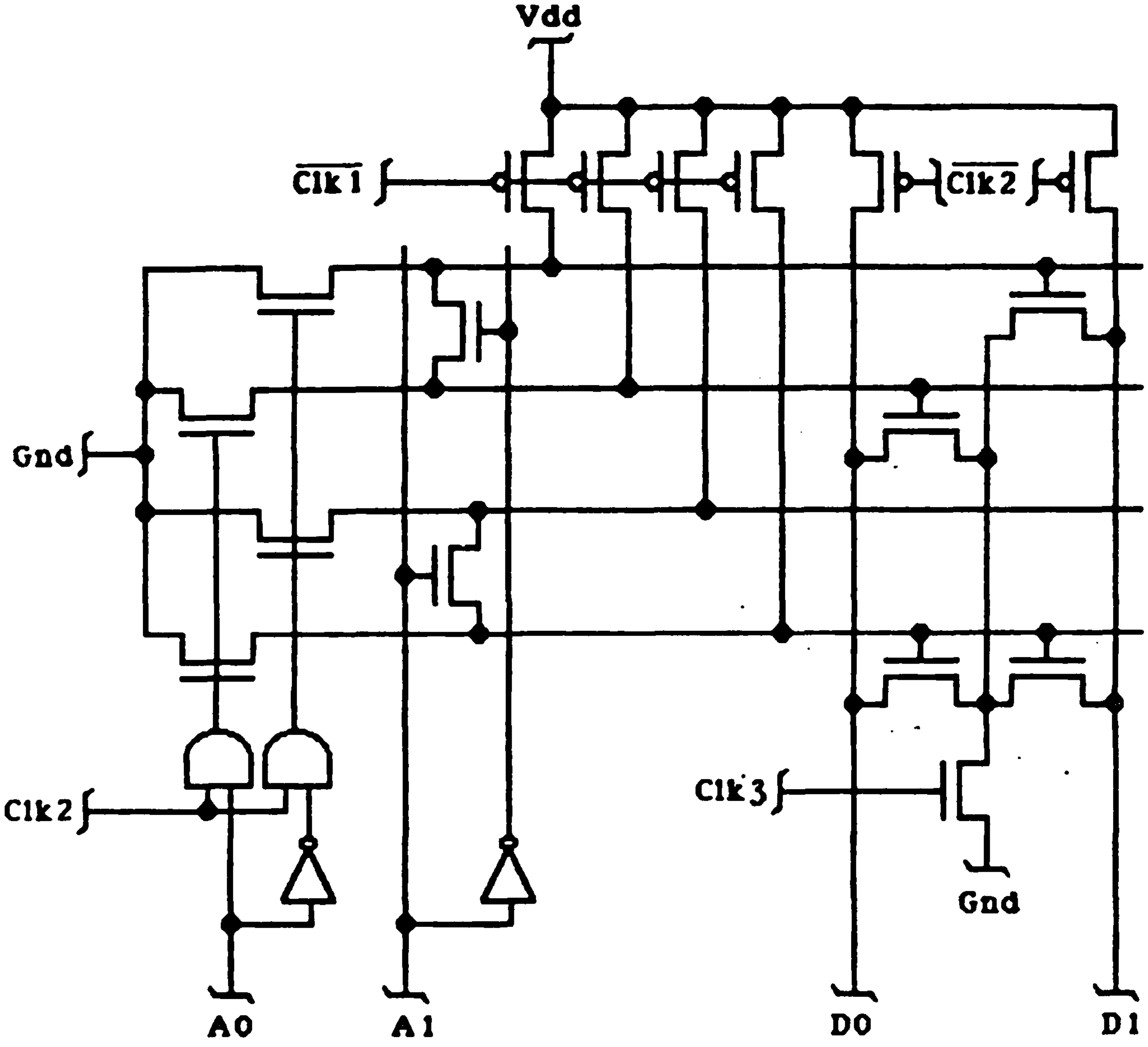


Figure 3

Multiple Clocks 4x2 Bit ROM:
(3 Phase)



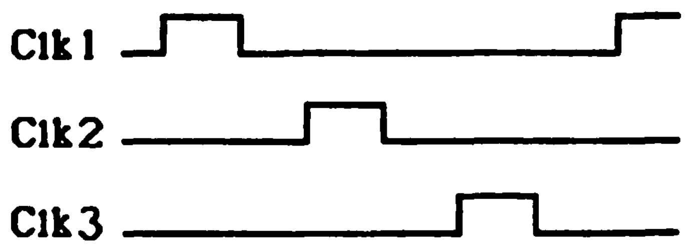
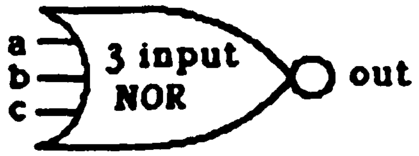
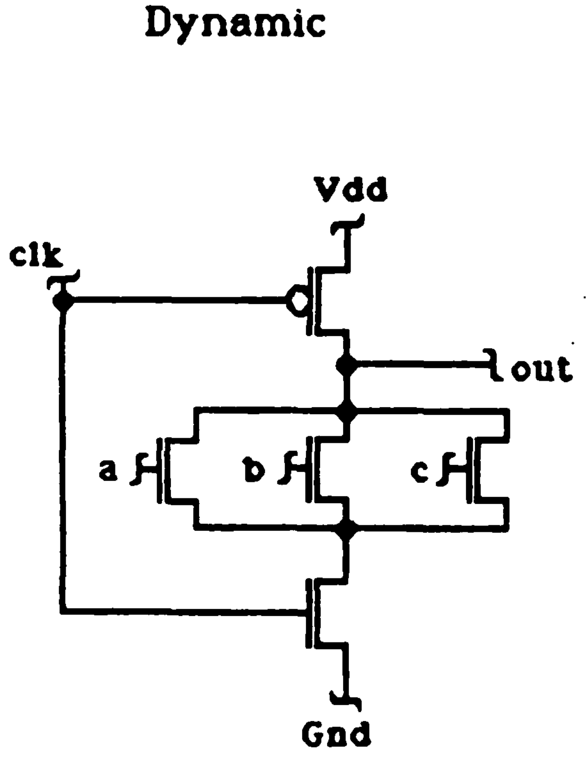
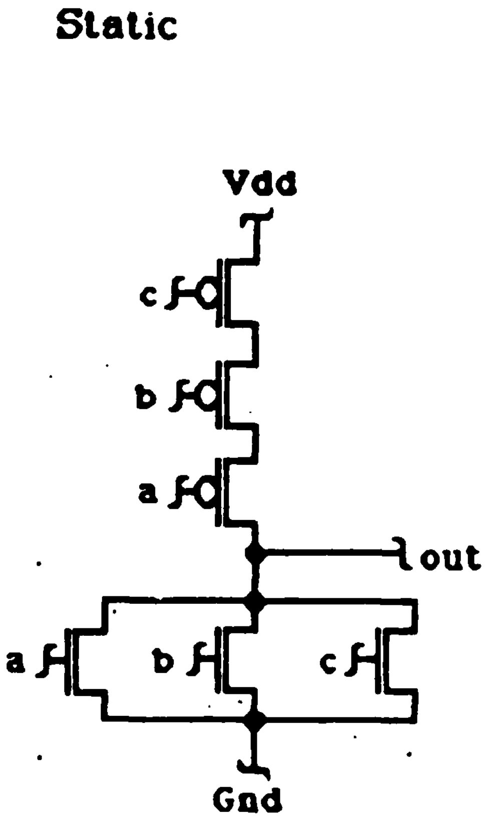


Figure 2

Dynamic versus Static Logic:







Transistors =
2 + Number of Inputs

Transistors =
2 x Number of Inputs

Figure 1

Modified Schichman-Hodges equations:

Transistor State:	**Current-Voltage Relationship:**
Off $(V_{gs} < V_{th})$	$i_{ds} = 0$
Saturation $(V_{gs} - V_{th} \leq (1+\delta)V_{ds})$	$i_{ds} = \frac{K_p S (V_{gs} - V_{th})^2}{2(1+\delta)}$
Linear $(V_{gs} - V_{th} \geq (1+\delta)V_{ds})$	$i_{ds} = K_p S \left[V_{ds}(V_{gs} - V_{th}) - \frac{(1+\delta)}{2} V_{ds}^2 \right]$

Where: $V_{th} = V_{to} + \gamma\left(\sqrt{V_{sb} + 2\phi_f} - \sqrt{2\phi_f}\right) \quad \delta = \frac{\gamma}{2(V_{sb}+2\phi_f)^{\frac{1}{2}}}$

Appendix B: Resistance Integrals

Drain connect loads:

(p-channel pulling high, n-channel pulling low)

The transistor is initially saturated and becomes linear as the load is discharged. This requires the use of 2 integrals to calculate the effective resistance.

$$R_{effective} = \int_{\frac{V_{gs}-V_{th}}{1+\delta}}^{V_{hi}} \frac{dV_{ds}}{i_{sat}} + \int_{V_{lo}}^{\frac{(V_{gs}-V_{th})}{(1+\delta)}} \frac{dV_{ds}}{i_{linear}} \tag{1}$$

Substituting for i_{sat} and i_{linear} as functions of V_{ds} and V_{gs} :

$$= \int_{\frac{V_{gs}-V_{th}}{(1+\delta)}}^{V_{hi}} \frac{dV_{ds}}{\frac{K_pS(V_{gs}-V_{th})^2}{2(1+\delta)}} + \int_{V_{lo}}^{\frac{V_{gs}-V_{th}}{(1+\delta)}} \frac{2dV_{ds}}{K_pS[2V_{ds}(V_{gs}-V_{th})-(1+\delta)V_{ds}^2]} \tag{2}$$

gives solutions of the form:

$$= \int_{C_2}^{C_1} g dx + \int_{C_3}^{C_2} \frac{dx}{ax(1-bx)} = gx \left|_{C_2}^{C_1} + (-\frac{1}{a}) ln \left| \frac{1-bx}{ax} \right| \right|_{C_3}^{C_2} \tag{3}$$

Where: $C_1 = V_{hi}$ $\quad C_2 = \frac{V_{gs}-V_{th}}{(1+\delta)}$ $\quad C_3 = v_{lo}$ $\quad g = \frac{2(1+\delta)}{K_pS(V_{gs}-V_{th})^2}$

$a = K_pS(V_{gs} - V_{th})$ $\quad b = \frac{(1+\delta)}{2(V_{gs}-V_{th})}$

$R_{effective} =$

$$\frac{2(1+\delta)}{KpS(V_{gs}-V_{th})^2}\left(V_{hi} - \frac{(V_{gs}-V_{th})}{(1+\delta)}\right)$$

$$+ \left(\frac{-1}{KpS(V_{gs}-V_{th})}\right) ln \left| \frac{1-\frac{1}{2}}{\frac{KpS(V_{gs}-V_{th})^2}{(1+\delta)}} \right| \tag{4}$$

$$- \left(\frac{-1}{KpS(V_{gs}-V_{th})}\right) ln \left| \frac{1-\frac{V_{lo}(1+\delta)}{2(V_{gs}-V_{th})}}{V_{lo}KpS(V_{gs}-V_{th})} \right|$$

Source connected loads:

(p-channel pulling low, or n-channel pulling high)

The transistor is <u>always</u> saturated as $V_{ds} \geq V_{gs}$.

$$R_{effective} = \int_{V_{lo}}^{V_{hi}} \frac{dV_s}{i_{sat}} = \int_{V_{lo}}^{V_{hi}} \frac{dV_s}{\frac{K_pS(V_g - V_s - V_{th})^2}{2(1+\delta)}} \tag{5}$$

Which gives a solution of the form:

$$= \int_{C_1}^{C_2} \frac{dx}{a(b-x)^2} = \int_{b-C_1}^{b-C_2} \frac{-du}{au^2} = \frac{1}{au}\bigg|_{b-C_1}^{b-C_2} \tag{6}$$

Where: $C_1 = V_{lo}$ $\quad C_2 = V_{hi}$ $\quad a = \frac{K_pS}{2(1+\delta)}$ $\quad b = V_g - V_{th}$ $\quad x = V_s$ $\quad u = b - x$

Therefore:

$$R_{effective} = \frac{2(1+\delta)}{K_pS}\left[\frac{1}{V_g - V_{th} - Vhi} - \frac{1}{V_g - V_{th} - V_{lo}}\right] \tag{7}$$